Objective

The objective of this workshop is to

a. Deploy an application and its associated database
b. Externalized all application configurations
c. Create a volume for the database
d. Create an Ingress resource to allow the application to be accessible from outside of the cluster

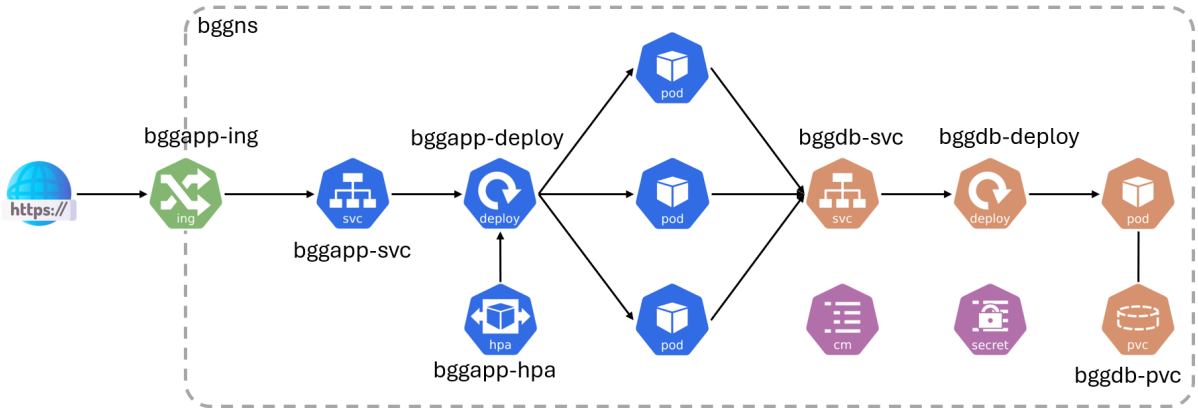
Setup

a. Assumed that you have a Docker Hub account. If not create one at https://hub.docker.com
b. This workshop will be using the following images
   i. Database – `chukmunnlee/bgg-database:nov-2025`
   ii. Application – `chukmunnlee/bgg-app:nov-2025`

Workshop

In this workshop you will be deploying a database (bgg-database) and an application (bgg-app) that uses the database.

The deployment is shown in the following diagram:



Database

Use a Deployment called `bggdb-deploy`, to deploy a single instance of the database. Provision a 1GB volume and mount it in the pod under `/var/lib/mysql`.

Create a service for the database pod. The service should only be accessible inside the cluster. The service port and the target ports are 3306.

The database user is `root` and its password can be changed by setting the environment variable `MYSQL_ROOT_DATABASE`.

The details of the database deployment are summarized as follows:

Resource	Name	Description
Persistent Volume Claim	`bggdb-pvc`	1 GB volume to be mounted under `/var/lib/mysql`
Deployment	`bggdb-deploy`	Deploy a single replica Use the image `chukmunnlee/bgg-database:nov-2025` Expose port 3306 Database user is `root` Database password is changed by setting `MYSQL_ROOT_PASSWORD`
Service	`bggdb-svc`	Service is only accessible from within the cluster only Service port is 3306

Application

Create a deployment called `bggapp-deploy` and deploy 3 instances of the image `chukmunnlee/bgg-app:nov-2025`.

The following environment variables is used to configure `bggapp-deploy`:

- `BGG_DB_USER` sets the database user viz. the user for login to the database. Use `root` to login to the database.
- `BGG_DB_PASSWORD` the database password. Use `mydbpassword` as the database password.
- `BGG_DB_HOST` is the database host
- `BGG_PORT` configures the port that the application listens on. The default is 5000.

Set the resource request and limit of each pod to be the following:

- Memory 96MB
- CPU 64 milli cores

Check the status of the application's pods by performing a HTTP GET on the `/healthz` resource.

Create a service called `bggapp-svc` to access the deployment. Set the service port to be the same as the target port. The service should only be accessible internally.

Expose the application with the name `bgg-<loadbalancer-ip>.sslip.io` with an Ingress called `bggapp-ing` On the

Provision horizontal auto scaler to scale this application according to the following requirements:

- Scale up to a maximum of 6 instances if the memory is above 80% utilization.
- Scale down to a minimum of 1 instance if the memory is below 20% utilization.

The following table summarizes the

Resource	Name	Description
Deployment	`bggapp-deploy`	Create 3 replicas Container port should be 5000 Probe the application's health on `/healthz`
Service	`bggapp-svc`	Service is only accessible from within the cluster only Service port should be the same as the target port
Horizontal Pod Autoscaler	`bggdb-hpa`	Scale `bggapp-deploy` to a maximum of 6 replicas when the memory utilization exceeds 80% Scale `bggapp-deploy` to a minimum of 1 replica1 when the memory utilization is below 20%
Ingress	`bggdb-ing`	Allow the application to be accessible on the host name `bgg-<loadbalancer-ip>.sslip.io`

Submission

Create a directory called `workshop03` inside your repo. Placed all the file for this workshop inside `workshop03` directory.

Push your work to GitHub after completing the workshop.